CS 221	Programming Languages	Homework Solution 1
Fall 2009		Due 13 Oct 2009

1. We use ref for the binary relation corresponding to the $reflect$ function.

$$\frac{\texttt{leaf}(n)}{\texttt{leaf}(n)\ \ \texttt{leaf}(n)\ \ \texttt{ref}} \quad (1)$$

$$\frac{\texttt{node}(l,r)\ \ \texttt{tree} \qquad r\ \ r2\ \ \texttt{ref} \qquad l\ \ l2\ \ \texttt{ref}}{\texttt{node}(l,r)\ \ \texttt{node}(r2,l2)\ \ \texttt{ref}} \quad (2)$$

We will say that "ref is single-valued at x" to mean that there is a unique y such that $x\ \ y$ ref and moreover that $y$tree. In this case, we call the corresponding y "the value of ref at x."

As rule (1) is the only rule which defines $x\ \ y$ ref with x a leaf, if leaf(n) holds, then there is a unique y such that leaf(n) y ref, namely $y =$ leaf(n).

Suppose inductively that r tree and l tree, and that ref is single-valued on r and l. Let $r2$ and $l2$ be the values of ref at r and l, respectively. Then by rule R^{tree}_{node} we know that node(l,r) and by rule (2), we know that node(l,r) node$(r2,l2)$ ref.

Suppose that node(l,r) node$(r2,l2)$ ref and also that node(l,r) node$(r3,l3)$ ref. Since rule (2) is the only rule that could have produced these expressions, we conclude that $r\ \ r3$ ref and $l\ \ l3$ ref. But by assumption ref is single-valued at r and l, hence we have $r2 = r3$ and $l2 = l3$, so ref is single-valued at node(l,r).

This completes the structural induction over the definition of a tree, hence we have shown that the binary relation ref as defined above is single-valued at every tree. In other words, ref well-defines a function, which we clearly recognize as the $reflect$ function defined in the exercise.

CS 221	Programming Languages	Homework Solution 2
Fall 2009		Due 13 Oct 2009

2. The rule

$$\frac{n \ \texttt{nat}}{n \ \texttt{succ}(n) \ \texttt{nat}} \quad (R^{less}_{incr})$$

is admissible but not derivable.

First, a useful lemma: n `nat` is derivable from no assumptions iff $n = \texttt{succ}^k(\texttt{zero})$ for some $k \in \mathbb{N}$. Obviously if $n = \texttt{succ}^k(\texttt{zero})$ then n `nat` is derivable from no assumptions by R^{nat}_{zero} and k applications of R^{nat}_{succ}. We show the converse by induction on the length of a derivation. If the derivation has length 1, then it must consist only of the rule R^{nat}_{zero}, in which case $n = \texttt{zero} = \texttt{succ}^0(\texttt{zero})$, as desired. If the derivation has length k, then it ends with the rule R^{nat}_{succ}, so we must have $n = \texttt{succ}(n_1)$ for some n_1 such that n_1 `nat` is derivable from no assumptions by a derivation of length $k - 1$. By induction, $n_1 = \texttt{succ}^{k-1}(\texttt{zero})$, hence $n = \texttt{succ}^k(\texttt{zero})$.

R^{less}_{incr} **is admissible:** Suppose n `nat` is derivable from no assumptions. By the lemma above, $n = \texttt{succ}^k(\texttt{zero})$ for some $k \in \mathbb{N}$. One application of R^{nat}_{zero} and one application of R^{less}_{zero} gives us

$$\texttt{zero} \ \texttt{succ}(\texttt{zero}) \ \texttt{less}$$

from no assumptions. Then k applications of R^{less}_{succ} gives us

$$\texttt{succ}^k(\texttt{zero}) \ \texttt{succ}^{k+1}(\texttt{zero}) \ \texttt{less}$$

which is easily seen to be equivalent to $n \ \texttt{succ}(n) \ \texttt{less}$.

R^{less}_{incr} **is not derivable:** If the rule were derivable, it would be true in every model of `nat`, `succ`. However, if we add the rule

$$\frac{}{\omega \ \texttt{nat}} \quad (R^{nat}_{\omega})$$

(where ω is a new primitive), we can show that $\omega \ \texttt{succ}(\omega) \ \texttt{less}$ is not derivable. For if ω ever appears in a statement of the form $x \ y$ `less`, it must appear as $\texttt{succ}(\omega)$. By induction on the derivation of such a statement: if the derivation uses R^{less}_{zero} applied to the conclusion of R^{nat}_{ω}, the conclusion is $\texttt{zero} \ \texttt{succ}(\omega) \ \texttt{less}$. If R^{less}_{succ} is used, then both arguments in the conclusion begin with `succ`. Hence $\omega \ \texttt{succ}(\omega) \ \texttt{less}$ is not derivable in this model, so R^{less}_{incr} is not derivable in general.

The two rules R^{less}_{zero} and R^{less}_{succ} are sufficient to define the usual less-than ordering on $\mathbb{N}$, as suggested by the lemma. Here is a derivation of $3 < 5$: from no assumptions, apply R^{nat}_{zero} and then R^{nat}_{succ} twice to get $\texttt{succ}^2(\texttt{zero}) \ \texttt{nat}$. Then apply R^{less}_{zero} to get $\texttt{zero} \ \texttt{succ}^2(\texttt{zero}) \ \texttt{less}$. Finally, apply R^{less}_{succ} three times to get the desired conclusion.

CS 221	**Programming Languages**	**Homework Solution 3**
Fall 2009		**Due 13 Oct 2009**

3. Suppose $s \overset{n}{\mapsto} s'$ for some $n \geq 0$. If $n = 0$, then we have $s' = s$, so we also have $s \overset{*}{\mapsto} s'$. If $n > 0$, then there must be an s'' such that $s \mapsto s''$ and $s'' \overset{n-1}{\mapsto} s'$. By induction on n, $s'' \overset{*}{\mapsto} s'$, and then applying the definition of $\overset{*}{\mapsto}$ we conclude that $s \overset{*}{\mapsto} s'$.

Conversely, suppose $s \overset{*}{\mapsto} s'$. Proceed by induction on the length of a derivation of this fact. If it can be derived in a single step, then $s' = s$, so we have $s \overset{0}{\mapsto} s'$. If it can be derived in $n > 1$ steps, then the last step must use the rule

$$\frac{s \mapsto s'' \quad s'' \overset{*}{\mapsto} s'}{s \overset{*}{\mapsto} s'}$$

and $s'' \overset{*}{\mapsto} s'$ must be derivable in $n-1$ steps. By the induction hypothesis, $s'' \overset{k}{\mapsto} s'$ for some $k \in \mathbb{N}$. Hence, applying the definition of $\overset{k+1}{\mapsto}$, we find that $s \overset{k+1}{\mapsto} s'$, as desired. $\square$

CS 221	**Programming Languages**	**Homework Solution 4**
Fall 2009		**Due 13 Oct 2009**

3. Lambda calculus

(a) Here λ, and . are terminals, as are all variable symbols x, where $x \in \mathrm{Var}$, a countable set of variable symbols. For instance Var could be the set of all alphanumeric identifiers.

$$\begin{array}{rcl} V & ::= & x, \ \ldots \quad (x \in \mathrm{Var}) \\ T & ::= & V \mid TT \mid \lambda V.T \end{array}$$

(b)

$$Terms \;\; t ::= \mathtt{var}[x] \mid \mathtt{apply}(t_1, t_2) \mid \mathtt{lambda}(\mathtt{var}[x], t)$$

where $x \in \mathrm{Var}$, and t, t_1, and t_2 all designate terms.

(c)

$$\frac{(x \in \mathrm{Var})}{\mathtt{var}[x] \;\; \mathtt{term}} \quad (\text{Variables})$$

$$\frac{t_1 \;\; \mathtt{term} \qquad t_2 \;\; \mathtt{term}}{\mathtt{apply}(t_1, t_2) \;\; \mathtt{term}} \quad (\text{Application})$$

$$\frac{\mathtt{var}[x] \;\; \mathtt{term} \qquad t \;\; \mathtt{term}}{\mathtt{lambda}(\mathtt{var}[x], t) \;\; \mathtt{term}} \quad (\lambda \text{ Abstraction})$$

(d)

$$\begin{array}{rcl} nlambdas(\mathtt{var}[x]) & = & 0 \\ nlambdas(\mathtt{apply}(t_1, t_2)) & = & nlambdas(t_1) + nlambdas(t_2) \\ nlambdas(\mathtt{lambda}(\mathtt{var}[n], t)) & = & 1 + nlambdas(t) \end{array}$$